Microsoft® SQL Server® 2008

Section 3

Your Data, Any Place,
Any Time

Data Retrieval

- To retrieve the information stored in the tables.

```
SELECT *
FROM tbl_students
```

```
SELECT s_id, s_name, s_dept
FROM tbl_students
```

```
SELECT s_name as Name,
S_dept department
FROM tbl_students
```

- Filtering with where clause

```
SELECT s_id,s_name
FROM tbl_students
Where s_dept=1
```

- Distinct

```
SELECT distinct s_dept
FROM tbl_students
```

Data Retrieval

- **Range Searching using BETWEEN – AND operator**

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept between 1 and 3
```

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept = 1 or
        s_dept = 2 or
          s_dept = 3
```

```
SELECT * FROM salesorder
WHERE orderdate BETWEEN '01/01/2005' AND
'06/30/2008'
```

Data Retrieval

- **IN predicate** searches for an exact match from a list.

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept in (1,7,8)
```

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept = 1 or
      s_dept = 7 or
        s_dept = 8
```

- **NOT IN predicate** searches for an not exact not match from a list.

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept in (1,7,8)
```

```
SELECT s_id, s_name
FROM tbl_students
Where s_dept != 1 and
      s_dept != 7 and
        s_dept != 8
```

Data Retrieval

- **Pattern Matching Using LIKE**

Description	SQL wildcard	Example
Multiple characters	%	'A%'
Single character	_	'_mangalore'
Any character within [ ]	[]	'[AO]%'
Not Any character within [ ]	[^]	'[^AO]%'

- List all employees whose name start with 'A'

```
SELECT ename FROM emp
WHERE ename LIKE 'A%'
```

- List all employees whose third character in name 'i'

```
SELECT ename FROM emp
WHERE ename LIKE '__i%'
```

Data Retrieval

- Top Rows

```
SELECT top 10 * FROM tbl_students
```

```
SELECT top 1 Percent FROM tbl_students
```

- Order by Clause

```
SELECT *
FROM tbl_students
WHERE s_dept=1
Order by s_name desc
```

```
SELECT top 10 *
FROM tbl_students
WHERE s_dept=1
Order by s_name desc
```

Retrieving Identity

- **SCOPE_IDENTITY()**
  - *Same session and the same scope.*
- **@@IDENTITY**
  - *Same session and across any scope.*
- **IDENT_CURRENT('TableName')**
  - *specify table across any table and any scope.*

SQL AGGREGATE FUNCTIONS

- SQL aggregate functions
  - return a single value, calculated from values in a column.
  - AVG() - Returns the average value
  - COUNT() - Returns the number of rows
  - FIRST() - Returns the first value
  - LAST() - Returns the last value
  - MAX() - Returns the largest value
  - MIN() - Returns the smallest value
  - SUM() - Returns the sum

SQL AGGREGATE FUNCTIONS

```
SELECT COUNT(*) FROM tbl_students
```

```
SELECT COUNT(*) FROM tbl_students
Where s_dept=1
```

```
SELECT COUNT (DISTINCT s_dept) FROM
    tbl_students
```

```
SELECT AVG(OrderQuantity) FROM Sales
    WHERE OrderPrice > 200
```

```
SELECT SUM(OrderPrice) FROM Sales
```

```
SELECT MAX(OrderPrice) FROM Sales
```

Group by clause

- The SQL GROUP BY statement
  - is used group a selected set of rows into a set of summary rows by the values of one or more columns or expressions.
  - It always used with one or more aggregate functions.

s_id	s_name	s_dept
1	H. Rizk	1
2	mohamed	2
5	ashraf	2
6	heba	5
7	manar	1
11	soha	3
17	majed	4

```
SELECT s_dept, COUNT(s_id) as [student count]
FROM tbl_students
GROUP BY s_dept
```

s_dept	student count
1	2
2	2
3	1
4	1
5	1

HAVING clause

- The HAVING clause
  - Is used to filter groups(after aggregation).
  - but WHERE clause is used to filter rows before aggregation can not be used with aggregate functions.

```
SELECT s_dept, COUNT(s_id) [student count]
FROM tbl_students
Where s_dept !=1
GROUP BY s_dept
```

VS

```
SELECT s_dept, COUNT(s_id) [student count]
FROM tbl_students
GROUP BY s_dept
having s_dept !=1
```

True Usage

```
SELECT s_dept, COUNT(s_id) [student count]
FROM tbl_students
Where s_dept !=1
GROUP BY s_dept
Having count(s_id)>1
```

Joins

- Joins
  - Are used to retrieve data from 2 or more related tables. In general tables are related to each other using foreign key constraint.
- Types of joins in sql server
  - Inner join
  - Outler join
    - Left outler join
    - Right outer join
    - Full outer join
  - Cross join

Join

- Inner join
  - returns all the matching rows between the two tables.

```
select s_name,d_name
from tbl_students
Inner join tbl_depts
on s_dept=d_id
```

- cross join
  - produces the Cartesian of the two tables. If table1 has 10 rows and table2 has 4 rows than the cross join produces 40 rows.

```
select s_name,d_name
from tbl_students Cross join tbl_depts
```

Join

- Left join
  - Returns all the matching rows +non matching rows from the left table.

```
select s_name,d_name
from tbl_depts Left outer join tbl_students
on s_dept=d_id
```

- Left join
  - Returns all the matching rows +non matching rows from the right table.

```
select s_name,d_name
from tbl_depts right outer join tbl_students
on s_dept=d_id
```

- Full join
  - Returns all the matching rows +non matching rows from both tables.

```
select s_name,d_name
from tbl_depts full outer join tbl_students
on s_dept=d_id
```

Join

- Can you retrieve the non matching rows from the left table?

```
select s_name,d_name
from tbl_students
left join tbl_depts
on s_dept=d_id
Where s_dept is null
```

- Can you retrieve the non matching rows from both tables?

```
select s_name,d_name
from tbl_students
Full join tbl_depts
on s_dept=d_id
Where s_dept is null
Or d_id is null
```

Self join

e_id	e_name	e_manager	e_salary
1	sayed mostafa	3	1000
2	Hamada Rizk	5	4000
3	Mohamed saad	4	2400
4	emad Etman	NULL	10000
5	amany sarhan	4	8000

Emp Name	Emp Salary	Manager name
sayed mostafa	1000	Mohamed saad
Hamada Rizk	4000	amany sarhan
Mohamed saad	2400	emad Etman
emad Etman	10000	NULL
amany sarhan	8000	emad Etman

```
select e.e_name as [Emp Name],
e.e_salary as [Emp Salary],
m.e_name as [Manager name]
from dbo.tbl_emps e
join dbo.tbl_emps m
on e.e_manager=m.e_id
```

```
select e.e_name as [Emp Name],
e.e_salary as [Emp Salary],
m.e_name as [Manager name]
from dbo.tbl_emps e
Left join dbo.tbl_emps m
on e.e_manager=m.e_id
```

- Self join
  - Joining table with itself.
  - Not a different type of join.
  - It can be any type of join:
    - Inner
    - Outer(left, right, full)
    - cross

Replacing NULL values

Emp Name	Emp Salary	Manager name
sayed mostafa	1000	Mohamed saad
Hamada Rizk	4000	amany sarhan
Mohamed saad	2400	emad Etman
emad Etman	10000	No Manager
amany sarhan	8000	emad Etman

Emp Name	Emp Salary	Manager name
sayed mostafa	1000	Mohamed saad
Hamada Rizk	4000	amany sarhan
Mohamed saad	2400	emad Etman
emad Etman	10000	NULL
amany sarhan	8000	emad Etman

```
ISNULL( m.e_name,'No Manager') as [Manager name]
```

```
COALESCE( m.e_name,'No Manager') as [Manager name]
```

```
CASE
WHEN m.e_name IS NULL THEN 'No Manager' ELSE m.e_name
END
```

```
select e.e_name as [Emp Name],
e.e_salary as [Emp Salary],
isnull( m.e_name,'No Manager') as
[Manager name]
from dbo.tbl_emps e
left join dbo.tbl_emps m
on e.e_manager=m.e_id
```


Thank You